

3-8-00

PTO/SB/05 (4/98)

Approved for use through 09/30/2000. OMB 0651-0032
 Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE

Please type a plus sign (+) inside this box → ☐

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

UTILITY PATENT APPLICATION TRANSMITTAL

(Only for new nonprovisional applications under 37 C.F.R. § 1.53(b))

Attorney Docket No. 584-1022

First Inventor or Application Identifier O'Doherty

Title Improved Session Initiation Protocol (SIP)

Express Mail Label No. EL 388 803 555 US

APPLICATION ELEMENTS

See MPEP chapter 600 concerning utility patent application contents.

ADDRESS TO: Assistant Commissioner for Patents
 Box Patent Application
 Washington, DC 20231

- ☒ * Fee Transmittal Form (e.g., PTO/SB/17)
 (Submit an original and a duplicate for fee processing)
- ☒ Specification [Total Pages 32]
 (preferred arrangement set forth below)
 - Descriptive title of the invention
 - Cross References to Related Applications
 - Statement Regarding Fed sponsored R & D
 - Reference to Microfiche Appendix
 - Background of the invention
 - Brief Summary of the invention
 - Brief Description of the Drawings (if filed)
 - Detailed Description
 - Claim(s)
 - Abstract of the Disclosure
- ☒ Drawing(s) (35 U.S.C. 113) [Total Sheets 6]
- Oath or Declaration [Total Pages 3]
 - ☒ Newly executed (original or copy)
 - ☐ Copy from a prior application (37 C.F.R. § 1.63(d))
 (for continuation/divisional with Box 16 completed)
 - ☐ DELETION OF INVENTOR(S)
 Signed statement attached deleting inventor(s) named in the prior application, see 37 C.F.R. §§ 1.63(d)(2) and 1.33(b).

- ☐ Microfiche Computer Program (Appendix)
- Nucleotide and/or Amino Acid Sequence Submission (if applicable, all necessary)
 - ☐ Computer Readable Copy
 - ☐ Paper Copy (identical to computer copy)
 - ☐ Statement verifying identity of above copies

ACCOMPANYING APPLICATION PARTS

- ☒ Assignment Papers (cover sheet & document(s))
- ☐ 37 C.F.R. § 3.73(b) Statement of Power of Attorney (when there is an assignee)
- ☐ English Translation Document (if applicable)
- ☒ Information Disclosure Statement (IDS)/PTO-1449 ☒ Copies of IDS Citations
- ☒ Preliminary Amendment
- ☒ Return Receipt Postcard (MPEP 503) (Should be specifically itemized)
- ☐ * Small Entity Statement(s) filed in prior application, Status still proper and desired (PTO/SB/09-12)
- ☐ Certified Copy of Priority Document(s) (if foreign priority is claimed)
- ☒ Other: Provisional Applications

* NOTE FOR ITEMS 1 & 13: IN ORDER TO BE ENTITLED TO PAY SMALL ENTITY FEES, A SMALL ENTITY STATEMENT IS REQUIRED (37 C.F.R. § 1.27), EXCEPT IF ONE FILED IN A PRIOR APPLICATION IS RELIED UPON (37 C.F.R. § 1.28).

16. If a CONTINUING APPLICATION, check appropriate box, and supply the requisite information below and in a preliminary amendment:

☐ Continuation ☐ Divisional ☐ Continuation-in-part (CIP) of prior application No. _____

Prior application information: Examiner _____ Group / Art Unit: _____

For CONTINUATION or DIVISIONAL APPS only: The entire disclosure of the prior application, from which an oath or declaration is supplied under Box 4b, is considered a part of the disclosure of the accompanying continuation or divisional application and is hereby incorporated by reference. The incorporation can only be relied upon when a portion has been inadvertently omitted from the submitted application parts.

17. CORRESPONDENCE ADDRESS

☐ Customer Number or Bar Code Label ☒ Correspondence address below
 (Insert Customer No. or Attach bar code label here)

Name William M. Lee, Jr.
 Lee, Mann, Smith, McWilliams, Sweeney & Ohlson
 Address P.O. Box 2786
 City Chicago State Illinois Zip Code 60690-2786
 Country USA Telephone (312) 368-1300 Fax (312) 368-0034

Name (Print/Type) William M. Lee, Jr. Registration No. (Attorney/Agent) 26,935
 Signature [Signature] Date 3/7/00

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Washington, DC 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Assistant Commissioner for Patents, Box Patent Application, Washington, DC 20231.

Improved Session Initiation Protocol (SIP)

FILING DATE: Herewith

1. Utility Patent Application Transmittal
2. Fee Transmittal for FY 1999 in duplicate
3. Assignment Transmittal and Assignment
4. Declaration and Power of Attorney
5. Amendment Accompanying Application
6. Form PTO-1449 with attached cited reference
7. Copies of Provisional Applications
8. Check No. 39029 for \$1432.00
9. Specification with six sheets of drawings
10. Certificate of EXPRESS MAIL.
11. Return Post Card.

DATE SENT: March 7, 2000
WJR:lmb

"Express Mail" mailing label number

EL 388 803 555 US

Date of deposit: March 7, 2000

I hereby certify that this paper or fee is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to the Commissioner of Patents and Trademarks, Washington, D.C. 20231.

Richard Sensenbrenner

(Typed or printed name of person mailing paper or fee)

Richard Sensenbrenner
(Signature of person mailing paper or fee)

584-1022

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

IN RE THE APPLICATION OF)

Michael O'Doherty)

SERIAL NO.: To be Assigned)

FILED: Herewith)

FOR: Improved Session Initiation Protocol (SIP))

AMENDMENT ACCOMPANYING APPLICATION

Honorable Commissioner of
Patents and Trademarks
Washington, D.C. 20231

Dear Sir:

It is requested that the application be amended as follows:

In the Specification

Page 1, after the title, add the following:

- - Related Application

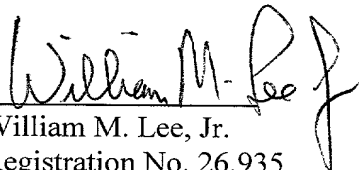
This application is the non-provisional filing of provisional applications numbers 60/171,777, filed December 22, 1999 and 60/171,801, filed December 22, 1999. - -

Remarks

The above amendment is being made in order to identify the provisional applications, upon which this application is based.

March 7, 2000

Respectfully submitted,


William M. Lee, Jr.
Registration No. 26,935
Lee, Mann, Smith, McWilliams,
Sweeney & Ohlson
P.O. Box 2786
Chicago, Illinois 60690-2786
(312) 368-6620
(312) 368-0034 (fax)

00/000" 030700

Background of the Invention

5

10

Description of the prior art

15

20

25

Multimedia teleconferencing and other conference calls are a complicated service for an end user to use. Because of this complexity many conferences experience problems or interrupts as various users set up the calls incorrectly. Central manual operators are often required to set-up the calls and this is expensive.

Summary of the Invention

Modifications to SIP are made which significantly extend the functionality of SIP for example by allowing a service for automatically setting up multi-media conferences to be easily provided. SIP messages are associated with computer software code such as Java byte code, Java applets or mobile autonomous software agents. An example of a mobile autonomous agent is a Java mobile agent. This computer software code may be contained in the body of a SIP message or an address indicating where the computer software code is located is stored in the SIP message. SIP clients are arranged such that on receipt of a SIP message that has been associated with computer software code, that code is executed by a processor associated with the SIP client. For example, in the case that Java applets are contained in a SIP message these are executed by a Java Virtual Machine associated with the SIP client. If a Java mobile agent is contained in the SIP message this executes on a Java Mobile Agent Virtual Machine associated with the SIP client. In one example, such computer software code must always be executed by the processor associated with the SIP client before that SIP client carries out any other actions related to the SIP message. Preferably an indicator is put into the

header of a SIP message to indicate that it has been associated with computer software code, and SIP clients are arranged to detect the presence of such indicators. An application programming interface is created in order that the computer software code may control the SIP client and/or any processor associated with that SIP client. In one example, computer software code is associated with SIP messages in order that a service for automatically setting up multi-media conferences is provided.

According to an aspect of the present invention there is provided a method of communicating between a first and a second node in a communications network, each of said nodes comprising a SIP client, said method comprising the steps of:-

- associating computer software code with a SIP message;
- sending the SIP message from the first SIP client associated with the first node to the second SIP client associated with the second node; and
- executing the computer software using the second node.

This provides the advantage that the functionality of SIP is greatly increased. It is possible to associate computer software code such as Java applets or a Java Mobile Agent with a SIP message such that the code is executed by a receiving communications network node. For example, the code can be used to control the second node in order to provide services such as a service for automatically setting up a multimedia conference call.

Preferably said computer software code is added to the SIP message. This enables the code to be easily accessed by the second node.

Preferably, said step of associating computer software code with the SIP message comprises adding an address to the SIP message which indicates where the computer software is stored. This provides the advantage that the size of the SIP message is not greatly increased whilst at the same time allowing the second node easy access to the computer software code using the address.

Preferably the method described above further comprises the step of proceeding with any SIP process related to the SIP message. This provides the advantage that any SIP process related to the SIP message is not affected by the presence of the computer software code unless that code is intended to affect that process.

Advantageously, the second SIP client is arranged such that on receipt of a SIP message containing an indicator, the computer software code associated with the SIP message is executed by the second node before that second node carries out any other processes related to the SIP message. This provides the advantage that if the computer software code is designed to affect the SIP process associated with the SIP message, this is achieved.

According to another aspect of the present invention there is provided a communications network node comprising:

- a SIP client;
- an input arranged to receive SIP messages which may be associated with computer software code;
- a processor arranged such that in use, when a SIP message is received, any computer software code associated with that SIP message is executed by the processor.

This provides the advantage that SIP messages that have been associated with computer software code in order to extend the functionality of SIP are received and the computer software code executed. This enables the extra functionality provided by the computer software code to be implemented.

According to another aspect of the present invention there is provided a computer program arranged to control a communications network node, said node comprising a SIP client and a processor, said computer program being arranged to control the node such that if a SIP message is received by the SIP client, any

computer software code associated with the received SIP message is executed by the processor.

According to another aspect of the present invention there is provided a communications network comprising a plurality of communications network nodes each such node comprising:

- a SIP client;
- an input arranged to receive SIP messages which may be associated with computer software code; and
- a processor arranged such that in use, when a SIP message is received, any computer software code associated with that SIP message is executed by the processor.

This provides the advantage that a communications network capable of implementing the improved SIP protocol is provided.

According to another aspect of the present invention there is provided a method of setting up a conference call between two or more parties, each party comprising a SIP client and a host processor, said method comprising the steps of:

- associating computer software code with a SIP message;
- sending the SIP message to each of the parties;
- executing the computer software code at each of the host processors.

This provides the advantage that a conference call is quickly and easily set up. The set-up process is taken care of by the computer software associated with the SIP messages.

According to another aspect of the present invention there is provided a system for automatically setting up a conference call between two or more parties, each party comprising a SIP client and a host processor, said system comprising:- a processor for associating computer software code with a SIP message and to send that SIP message to each of the parties; and wherein each of said host processors is

arranged to execute the computer software code in use, when the SIP message is received. The system provides a means for automatically setting up conference calls such that attendees do not need to take complex actions to set up the call.

According to another aspect of the present invention there is provided a method
5 of upgrading or replacing interconnected SIP clients each SIP client being associated with a host processor said method comprising the steps of:-

- associating computer software code suitable for said upgrade or replacement with a SIP message;
- sending the SIP message to each of the SIP clients; and
- 10 • executing the computer software at each of the host processors.

This provides the advantage that a plurality of SIP clients that are connected, for example in a communications network, may be upgraded or replaced quickly and easily. The upgrade or replacement process may be automated and operators are not required to make the upgrade or replacement using CDs or other media carrying
15 the new software to each SIP client individually.

According to another aspect of the present invention there is provided a method of testing members of a group of SIP clients each SIP client being associated with a host processor said method comprising the steps of:-

- 20 • associating computer software code suitable for said testing with a SIP message;
- sending the SIP message one of the SIP clients;
- executing the computer software at the host processor associated with that SIP client in order to obtain test results; and
- repeating steps (ii) to (iii) for each of the other SIP clients in the group.

25 This provides the advantage that a group of SIP clients, for example, in a communications network, may be automatically tested in a quick and efficient manner. For example, if an error is reported on a network and the location or nature

of that error is unknown, this method of testing may be used to investigate the situation.

According to another aspect of the present invention there is provided a method of forwarding a call from a first SIP client to a second SIP client, each of said SIP clients being associated with a host processor, said method comprising the steps of:-

- receiving a call at the first SIP client and if that call is not answered then associating computer software code with a SIP message said computer software code being arranged to forward a call;
- sending the SIP message from the first SIP client to a specified second SIP client; and
- executing the computer software using the host processor associated with the second SIP client such that the call is forwarded to the second SIP client.

This provides the advantage that a call is quickly and efficiently forwarded in the event that the call is not answered at a first SIP client. This method can be extended for greater numbers of connected SIP clients, for example, so that if a call to a person in an office is not answered that call will automatically be forwarded to other terminals in the office until the call is answered.

Further benefits and advantages of the invention will become apparent from a consideration of the following detailed description given with reference to the accompanying drawings, which specify and show preferred embodiments of the invention.

Brief description of the drawings

Figure 1 is a schematic diagram of a communications network which incorporates nodes for implementing an improved SIP protocol.

Figure 2 is a flow diagram of a method of communicating between two SIP clients using an improved SIP protocol.

Figure 4 shows the format of an improved SIP protocol message.

5 Figure 6 is a flow diagram of a method of setting up a conference call using a conference call service system.

Figure 8 shows a method of upgrading or replacing interconnected SIP clients.

10 Figure 10 shows a method of forwarding a call from a first SIP client to a second SIP client.

Embodiments of the present invention are described below by way of example only. These examples represent the best ways of putting the invention into practice that are currently known to the Applicant although they are not the only ways in which this could be achieved.

The term "Java virtual machine" is used to refer to a processor which is arranged to execute Java applets or Java byte code.

The term “mobile autonomous software agent” is used to refer to a computer program that is able to halt itself and move itself from a first processor to another processor that is connected to the first processor for example by a communications network. The computer program is referred to as being autonomous because it is able to “decide” where to move and what it will do independently of external

requests. An example of a mobile autonomous software agent is a Java mobile agent. Details about Java mobile agents are given in the article, "Under the Hood: The architecture of aglets", by Bill Venners, JavaWorld April 1997 the contents of which are incorporated herein by reference.

By extending the SIP protocol increased functionality is provided. SIP messages are modified to carry computer software code such as Java applets or to carry an address such as an universal resource locator (URL) indicating where computer software code is stored. An application programming interface (API) is also defined which allows the computer software code to interact with a receiving host system. SIP clients are also modified in order that they execute the computer software code associated with the SIP messages before any other actions are taken as a result of receipt of the SIP message.

Figure 1 shows a communications network 1 comprising a plurality of communications network nodes 10 each such node comprising:

- 15 • a SIP client 11;
- an input 12 arranged to receive SIP messages which may be associated with computer software code; and
- a processor 13 arranged such that in use, when a SIP message is received, any computer software code associated with that SIP message is executed by the
- 20 processor. This processor is provided by the host system and may comprise a Java virtual machine or any other suitable processor. These communications network nodes are referred to as enhanced SIP nodes because they are arranged to allow the enhanced SIP process to work.

25 The communications network of Figure 1 is used in conjunction with the method illustrated in Figure 2 in order to implement the enhanced SIP process. Figure 2 is a flow diagram of a method of communicating between a first and a second node in a

communications network, each of said nodes comprising a SIP client, said method comprising the steps of:-

- associating computer software code with a SIP message (box 20 in Figure 2);
- sending the SIP message from the first SIP client associated with the first node to
- 5 the second SIP client associated with the second node (box 21 in Figure 2); and
- executing the computer software using the second node (box 22 in Figure 2).

For example, Figure 3 illustrates an example of how a plurality of enhanced SIP clients 30, 31, 32, 33, 41 interact. Each SIP client is supported on a communications

10 network node (not shown). SIP client A 30 is connected to SIP client B 31 via a communications link 34 and SIP client B 31 is connected to both SIP client C 32 and SIP client D 33 via communications links 34. SIP client B 31 has a host system 35 which comprises a Java virtual machine. SIP client D 33 is also connected to SIP client E via a communications link. SIP client D and has a host system 39 which

15 comprises a Java mobile agent virtual machine and SIP client E 41 also has a host system 41 which comprises a Java mobile agent virtual machine 42.

Using the enhanced SIP protocol, computer software code such as Java applets are associated with a SIP message 36. That is, the computer software code may be added to the SIP message body itself or may be stored separately and an address of

20 the storage location added to the SIP message. It is not essential to use Java applets or Java mobile agents; any other suitable computer software code may be used. The message 36 is sent from SIP client A 30 to SIP client B 31. SIP client B detects the presence of the Java applets (or other computer software code) associated with the SIP message 36 and executes these Java applets using its Java

25 virtual machine 35 (or other type of host processor).

Any suitable method of detecting the presence of computer software code associated with the SIP message 36 may be used. For example, an indicator may

be placed in the header of the SIP message 36 and the SIP client 31 arranged to detect that indicator and associate it with the presence of computer software code. An example of such an indicator in a SIP message is described in more detail below.

By executing the Java applets, two new SIP messages 37, 38 are created one of which 37 contains a Java mobile agent and the other which does not. This is just one example of a something that the computer software code associated with the SIP message could do. For example, the computer software code could also be arranged to modify existing SIP messages, delete existing SIP messages, generate SIP messages, receive SIP messages or to control the SIP client and/or the host processor to perform any other suitable function. The computer software code is arranged to interact with the host processor via an API as described below. Security restrictions may be enforced by the SIP client and or host system in order to limit the actions that any software code associated with a SIP message is able to effect. More detail about these security restrictions is given below.

The executed Java applets then cause SIP client B 31 to send one of the created messages 37 to SIP client D 33 and the other 38 to SIP client C 32. The message 37 sent to SIP client D contains a Java mobile agent (or other computer software code or an address of computer software code). If SIP client D has the capability to execute the Java mobile agent contained in message 37 then SIP client D does so. However, if SIP client D does not have this capability, for example, if SIP client D has no Java mobile agent virtual machine, then SIP client D simply follows the standard SIP procedure for unsupported require extensions. This involves returning an error message to SIP client B, indicating that the Java applet in message 37 was not executed.

In the meantime, SIP message 38 which is not associated with any computer software code, is sent to SIP client C 32 and any SIP process associated with that message 38 is carried out following the standard SIP protocol.

In this example, SIP client D does have an associated Java mobile agent virtual machine 39 and so when message 37 arrives, the Java mobile agent in message 37 begins to execute on this processor. At some point in the execution, the Java mobile agent suspends itself and includes itself in SIP message 40 which is sent to SIP client E. This is one example of a process that may occur by incorporating a Java mobile agent into a SIP message.

In the enhanced SIP protocol described herein, standard SIP messages are modified by associating computer software code with them as described above. For example, one or more Java applets or Java mobile agents are stored in a multipart MIME section in the body of a SIP message or a URL indicating where the Java applets or Java mobile agents are stored is added to the SIP message.

In some examples, an indicator is added to the SIP message header, in order to indicate that computer software code is associated with that SIP message. For example, a "Require request-header" is used to indicate that Java enhanced SIP must be supported to process a SIP message that is associated with Java applets or Java byte code. This require request header is the same as the header for a standard SIP message except that the content type field in the entity header is used to indicate that the content type is a Java applet or the URL of a Java applet which must be retrieved. Also, the require field of the request-header is used to specify that Java enhanced SIP must be supported to process the message concerned.

Figure 4 illustrates the structure of a standard SIP message and shows how this structure is used in the improved SIP protocol described herein. The structure of a standard SIP message is illustrated at 40 in Figure 4. Thus a standard SIP message comprises a general-header, a request-header, an entity header, a CRLF and a message body. The structure of a general-header is shown at 41 in Figure 4 and similarly the structures of each of an entity header 42, request header 43 and response header 44 are shown. In order to indicate that the improved SIP protocol described herein is being used markers or tags are included in the SIP message in

any suitable location. For example, the content-type field of an entity header may be used to indicate that the content type is a Java applet or the URL of a location of a Java applet. Similarly, the content-type field of an entity header may be used to indicate that the content type is a Java mobile agent or the URL of a location of a Java mobile agent. Also, the require field of a request header may be used to indicate that Java enhanced SIP must be supported to process the message concerned. However, it is not essential to use the content-type field or the require field for this purpose. Any other suitable field(s) may be used.

Figure 5 shows an example of an INVITE message according to the improved SIP protocol described herein. The content type field contains the words "multipart /mixed" which indicates that the INVITE message body is in the form of a MIME multipart message which contains one or more Java applets or Java mobile agents. The require field contains the words "org.ietf.sip.java-enhanced-sip" which indicate that the improved SIP protocol must be used to process this message. Part of the body of the INVITE message containing the Java applet(s) or Java mobile agents is shown 50.

The SIP clients used to implement the improved SIP protocol are the same as standard SIP clients except that they are arranged to do the following things:

- Detect improved SIP messages which are associated with computer software code. For example, this may be done by arranging the SIP client to recognise the presence of the words "org.ietf.sip.java-enhanced-sip" or "org.ietf.sip.java-mobile-agent-enhanced-sip" in the SIP message header.
- If an improved SIP message is received and detected, the software code associated with that SIP message is accessed by the SIP client and executed on the SIP client's host processor. Preferably, this execution is carried out immediately, before processing the SIP message any further. For example, if a content type field in a SIP header indicates that a URL for a Java applet is

present then the SIP client must immediately get the applet from the URL and execute the applet on a Java virtual machine associated with the SIP client. If the SIP client does not execute the software code then it is preferably arranged to respond by returning status code 420 (bad extension) and by listing
 5 org.ietf.sip.java-enhanced-sip in an unsupported header. The SIP client may not execute the software code if it is unable to do so, for example, if no Java virtual machine is available, or if the SIP client decides not to do this , for example, for security reasons.

- Match incoming SIP messages to patterns and in the event of a match “wake up”
 10 any waiting computer software code. This is described in more detail below.

The SIP client's host processor is modified as compared to a standard SIP client's host processor in that it must comprise a processor of a specific type. For example, a Java virtual machine in the case that Java applets are associated with
 15 the improved SIP messages. In the case that Java mobile agents are used, a Java mobile agent virtual machine is required. Also, the SIP client's host processor has access to or comprises an API to allow the computer software code associated with the improved SIP messages to interact with the SIP client. For example, in the case that Java applets are used, the SIP client's host has access to a set of Java classes
 20 or applets that are defined in a Java enhanced SIP API. This API allows access into the SIP client to allow SIP messages to be built and sent subject to security restrictions. Using the API received Java applets or Java mobile agents are able to generate and receive SIP messages using the receiving SIP client.

25 Passing of control between the computer software code associated with improved SIP messages and the SIP client concerned.

In the case that standard SIP messages are used, these are processed by SIP clients in the standard way and control remains with the SIP clients. However, in

the improved SIP case described herein, any computer software code associated with a SIP message takes precedence over other standard SIP processes associated with the SIP message or with any other SIP messages received by a SIP client during processing of the computer software code.

5 For example, the computer software code associated with a SIP message can be arranged to initiate a SIP session and to wait for a SIP response before proceeding. During this waiting period, control remains with the computer software code. The computer software code is able to specify that it will go to sleep and wait for the next SIP message which matches a particular pattern. In that case, the SIP
10 client does no other actions during the sleep period. Alternatively, the computer software code can deal with any other incoming SIP messages itself during the sleep period. Thus control does not pass back to the SIP client until the computer software code wants it to even if SIP messages from other sessions are arriving.

15 Application programming interface (API)

As described above an API is specified in order that the computer software code associated with improved SIP messages is able to affect the SIP client. For example, this API allows a received Java applet or Java mobile agent access to the SIP messaging functions on the SIP client.

20 Examples of methods that the API supports comprise:

- SendSIPMessage – sends a SIP message and establishes a context for the Session if one does not already exist. The invoker (which is the piece of software code which called this function) can indicate if it wants the message to be part of an existing Session. For example, the invoker could be a Java applet or Java
25 mobile agent.
- ReceiveSIPMessage – retrieves a SIP message from the Client's input buffer on a first in first out (FIFO) basis.

- ReceivedMessageSummary – returns a summary of any received messages in the client's input buffer along with a count of messages received. If the client does not support buffering of input messages this is indicated.
- QueryCapabilities – returns the capabilities of the Client. These include the ability to buffer incoming messages and the buffer size.
- Querystatus – returns the status of any sessions the client is currently involved in.
- MatchMessageAndWake – checks incoming messages against a particular pattern and if they match wakes up the indicated applet or Java mobile agent and passes the messages directly to the indicated applet.
- ProcessMessage – sends a message to the Client and passes control to the client for the message to be processed as in standard SIP. For example, this can be used after an applet or Java mobile agent has looked at the message or altered it in some way and then wants to pass the message back to the client to be processed as in standard SIP.
- ProcessMessageAndReturn – as for ProcessMessage except that control is passed back to the invoker after the message has been processed.
- ProcessFromBufferAndReturn – processes the next message on the INPUT buffer as in standard SIP within the client and then returns control to the invoking applet or Java mobile agent.

Changes to SIP proxy and SIP server behaviour

Following standard SIP as defined in "Request for comments (RFC) 2543 SIP: Session Initiation Protocol", SIP proxy and redirect servers must ignore features that are not understood. That is, if a SIP proxy or redirect server is not arranged to understand the improved SIP messages described herein then it must ignore features of those messages that are not common to standard SIP. A SIP proxy server is a communications network node which communicates using the SIP

5

Security

10

An example of an algorithm for a security mechanism is:

- Index the matrix for user defined security checks against that operation

- Extract the method corresponding to the security action datafilled by the user
 - Execute that security mechanism method
 - If the result of the security mechanism method is “pass” then continue and call the SIP API method
- 5 • Else display a security disallowed message and return without calling the SIP API method.

Actions that a user may datafill for a given SIP operation include:

- Allow always
 - Disallow always
- 10 • Allow conditional
- Disallow conditional
 - Prompt y/n
 - Allow and display warning or info

An example of use of the improved SIP protocol to create a service for automatically setting up multimedia conferences is now described.

15

Conferencing system

Using the improved SIP protocol a conferencing service is created whereby a single chairperson is able to set up the conference by sending out SIP INVITE messages. The method is suitable for multimedia conferences. The INVITE messages are associated with computer software code which executes on the host machines of invited attendees to set up the conference call. This greatly simplifies the process of setting up a conference call such as a multimedia conference call.

20

For example, the computer software code associated with the improved SIP INVITE messages can be arranged to set up connections from each attendee's machine to several video sources and to an electronic whiteboard to be shared for the meeting. The computer software code can also be arranged to start up a web browser to a page relevant to the meeting on each attendee's machine. As well as

25

5

10

15

20

25

audio, video and data streams associated with the conference as appropriate given the capabilities (box 63 of Figure 6). Depending on how the user has his or her security mechanisms set he or she may be prompted before the sessions are set up for the various media streams. When the Java applet(s) initiate the SIP sessions

5 (box 63 of Figure 6) they may also be arranged to set up these SIP sessions such that all the attendees except for a chairperson are on mute. This is particularly advantageous, because the chairperson is then easily able to announce the beginning of the meeting and to chair the meeting in an organised fashion.

The Java applets(s) may also be arranged to forward details of a web page

10 from each attendee to a chairperson or to the conference service system. For example, a web page giving biographical details of each attendee may be forwarded to a chairperson who then makes these available to each other attendee. In a similar manner, digital photographs of each attendee may be forwarded to the chairperson by the Java applets. It is also possible for the Java applets to request a joining

15 message from each attendee which is then forwarded to a chairperson automatically by the Java applets. This joining message may contain security requirements specific to each attendee.

Depending on the number of parties to the conference, a conferencing bridge facility may be used as is known in the art. Alternatively, a software based technique

20 is used to connect the parties to the conference.

An example of an algorithm that is encoded in the Java applet(s) of the method described immediately above is:

- Read the message that the Java applet was associated with to obtain the addresses for the various streams in the call
- 25 • Query the capabilities of the SIP client
- Query the capabilities of the host system

- 5

10

15

20

25

Hunt group system

An example of the use of improved SIP with Java mobile agents is now described. In this example, a service is provided whereby an automated system calls several telephones within a defined group (such as a team in an office) until one of those
5 telephones is answered. For example, the nodes of the communications network in Figure 1 may each provide a telephone implemented by software in the SIP clients 11. Each telephone within the group 1 comprises a SIP client 11 and a host
processor 13 as illustrated in Figure 1 and the telephones are connected to one another via a communications network 1 as shown in Figure 1. The host processors
10 each comprise a Java mobile agent virtual machine.

A user, which may be an automated service or a human using a terminal connected to the communications network 1, telephones one of the telephones 10 within the defined group. If the called telephone is not answered after a specified number of rings or an elapsed time, then software at the SIP client 11 of the called
15 telephone creates a Java mobile agent, associates this with a SIP message, and sends the SIP message to a predefined second SIP client. This second SIP client is one of the telephones within the defined group 1.

The second SIP client receives the SIP message which is associated with the Java mobile agent. The Java mobile agent then executes itself on the Java mobile agent virtual machine associated with the second SIP client. The Java mobile agent is arranged to apply ringing to the second telephone and queries the second telephone's identification details and sends these back to the original caller. If the caller is using a host processor that has a display system associated with it, then information about the call and the fact that it has been forwarded to the second telephone in the defined group is sent by the Java mobile agent to this display.

If the second SIP client does not answer after a specified number of rings or time then the second SIP client repeats the method that the first SIP client carried out

as described above. However, the second SIP client incorporates information about the fact that the call has been forwarded again.

After the method has been repeated a pre-determined number of times and if the call is not answered, then the call is sent back to the first SIP client that was
 5 called. A display of the route taken and the fact that the call was not answered is made at the first SIP client if a display is available.

If the call is answered, information about the route taken and the identity of the answering SIP client is sent back to the caller which may be an automated service.

10 Figure 10 shows a method of forwarding a call from a first SIP client to a second SIP client, each of said SIP clients being associated with a host processor, said method comprising the steps of:-

- receiving a call at the first SIP client and if that call is not answered then associating computer software code with a SIP message said computer software
 15 code being arranged to forward a call (box 100 Figure 10);
- sending the SIP message from the first SIP client to a specified second SIP client (box 101 Figure 10); and
- executing the computer software using the host processor associated with the second SIP client such that the call is forwarded to the second SIP client (box
 20 102 Figure 10).

Client test system

Another example of the use of Java mobile agents with improved SIP involves a test system for a pre-defined group of SIP clients. For example, the network of SIP
 25 clients shown in Figure 1. The SIP clients 11 are connected to one another to form a communications network 1 as illustrated in Figure 1. Each SIP client 11 is

associated with a host processor 13 which comprises a Java mobile agent virtual machine.

- A test system (for example, software located at one of the nodes 10 in the communications network 1), which may be an automated software service, creates a
- 5 Java mobile agent, associates this with a SIP message, and sends that SIP message to one of the SIP clients 11 in the group. The Java mobile agent executes on the receiving SIP client and sets up one or more test sessions. The results of these test sessions are stored by the Java mobile agent in its private data, together with any other required information. The Java mobile agent then associates itself with another
- 10 SIP message and arranges that this SIP message be sent to another SIP client in the group. When the SIP message reaches another SIP client the process of obtaining information is repeated so that more information is added to the Java mobile agent's private data. Another SIP message is used to send the Java mobile agent on to another SIP client and so on, until all the SIP clients in the group have been visited.
- 15 Once all the SIP client's in the group have been visited by the Java mobile agent, this agent associates itself with a SIP message in order to be sent back to the originating SIP client. In this way the Java mobile agent is able to report the results of its tests to the originating SIP client. The Java mobile agent may also be arranged to initiate other actions to fix any faults that it finds as it finds them.
- 20 Figure 9 shows a method of testing members of a group of SIP clients each SIP client being associated with a host processor said method comprising the steps of:-
- associating computer software code suitable for said testing with a SIP message (box 90 Figure 9);
 - sending the SIP message one of the SIP clients (box 91 Figure 9);
 - 25 • executing the computer software at the host processor associated with that SIP client in order to obtain test results (box 92 Figure 9); and

- repeating steps (ii) to (iii) for each of the other SIP clients in the group (box 93 Figure 9).

Upgrade or replacement of SIP clients

- 5 Consider a situation in which it is required to upgrade or replace SIP clients which support the improved version of SIP described herein. This may be carried out automatically as follows:

The software for the upgrade or new SIP client is associated with a SIP message, for example, by building the software into a Java applet and adding this applet to a SIP message. This SIP message is then sent to all the SIP clients which are to be upgraded or replaced. On receipt of the SIP message at a SIP client, the existing SIP client runs the software code in order to effect the upgrade or replacement. The extent to which the upgrade or replacement is effected depends on the security specifications and the type of SIP client. By using the improved SIP protocol in this way, upgrades or replacement of a plurality of SIP clients is achieved quickly and easily.

Figure 8 shows a method of upgrading or replacing interconnected SIP clients each SIP client being associated with a host processor said method comprising the steps of:-

- 20
- associating computer software code suitable for said upgrade or replacement with a SIP message (box 80 Figure 8);
 - sending the SIP message to each of the SIP clients (box 81 Figure 8); and
 - executing the computer software at each of the host processors (box 82 Figure 8).

25 A range of applications are within the scope of the invention. These include situations in which it is required to communicate between entities using an improved SIP protocol. For example to enable multimedia conferences to be set up

5
10

11. A method as claimed in claim 1 which further comprises adding an indicator to a header of the SIP message in order to indicate the presence of the computer software code and arranging the second SIP client to recognise the indicator.
- 5 12. A method as claimed in claim 1 which further comprises the step of proceeding with any SIP process related to the SIP message.
13. A method as claimed in claim 11 wherein said second SIP client is arranged such that on receipt of a SIP message containing such an indicator, the computer software code associated with the SIP message is executed by the
10 second node before that second node carries out any other processes related to the SIP message.
14. A method as claimed in claim 1 wherein said computer software is arranged to interact with the second SIP client via a specified application programming interface.
- 15 15. A method as claimed in claim 1 wherein said computer software is arranged to interact with the processor associated with the second SIP client via a specified application programming interface.
16. A method as claimed in claim 1 wherein said computer software code is arranged to set up a multimedia conference call.
- 20 17. A method as claimed in claim 1 wherein said computer software code is arranged to upgrade or replace said SIP client.
18. A method as claimed in claim 1 wherein said computer software code is arranged to test said second node.
19. A method as claimed in claim 1 wherein said computer software code is
25 arranged to forward a call from the first to the second node.
20. A communications network node comprising:
 - (i) a SIP client;

- (ii) an input arranged to receive SIP messages which may be associated with computer software code;
 - (iii) a processor arranged such that in use, when a SIP message is received, any computer software code associated with that SIP message is executed by the processor.
- 5
21. A communications network node as claimed in claim 15 wherein said processor comprises a Java virtual machine.
22. A communications network node as claimed in claim 15 which further comprises an application programming interface arranged to allow the
- 10 computer software code to interact with the SIP client.
23. A communications network node as claimed in claim 15 wherein said processor further comprises a detector arranged to detect an indicator in a received SIP message which indicates that computer software code is associated with that SIP message.
- 15 24. A computer program arranged to control a communications network node, said node comprising a SIP client and a processor, said computer program being arranged to control the node such that if a SIP message is received by the SIP client, any computer software code associated with the received SIP message is executed by the processor.
- 20 25. A computer program as claimed in claim 24 which is stored on a computer readable medium.
26. A communications network comprising a plurality of communications network nodes each such node comprising:
- (i) a SIP client;
 - (ii) an input arranged to receive SIP messages which may be associated with computer software code; and
- 25

- (iii) a processor arranged such that in use, when a SIP message is received, any computer software code associated with that SIP message is executed by the processor.
27. A method of setting up a conference call between two or more parties, each party comprising a SIP client and a host processor, said method comprising the steps of:
- (i) associating computer software code with a SIP message;
- (ii) sending the SIP message to each of the parties;
- (iii) executing the computer software code at each of the host processors.
28. A method as claimed in claim 27 wherein the computer software code is arranged to take into account capabilities of each host processor.
29. A method as claimed in claim 27 wherein said conference call is a multimedia conference call.
30. A system for automatically setting up a conference call between two or more parties, each party comprising a SIP client and a host processor, said system comprising:- a processor for associating computer software code with a SIP message and to send that SIP message to each of the parties; and wherein each of said host processors is arranged to execute the computer software code in use, when the SIP message is received.
31. A method of upgrading or replacing interconnected SIP clients each SIP client being associated with a host processor said method comprising the steps of:-
- (i) associating computer software code suitable for said upgrade or replacement with a SIP message;
- (ii) sending the SIP message to each of the SIP clients; and
- (iii) executing the computer software at each of the host processors.
32. A method of testing members of a group of SIP clients each SIP client being associated with a host processor said method comprising the steps of:-

- (i) associating computer software code suitable for said testing with a SIP message;
- (ii) sending the SIP message one of the SIP clients;
- (ii) executing the computer software at the host processor associated with that SIP client in order to obtain test results; and
- (iii) repeating steps (ii) to (iii) for each of the other SIP clients in the group.
33. A method of forwarding a call from a first SIP client to a second SIP client, each of said SIP clients being associated with a host processor, said method comprising the steps of:-
- (i) receiving a call at the first SIP client and if that call is not answered then associating computer software code with a SIP message said computer software code being arranged to forward a call;
- (ii) sending the SIP message from the first SIP client to a specified second SIP client; and
- (iii) executing the computer software using the host processor associated with the second SIP client such that the call is forwarded to the second SIP client.

ABSTRACT**Improved session initiation protocol (sip)**

Modifications to SIP are made which significantly extend the functionality of SIP for example by allowing a service for automatically setting up multi-media

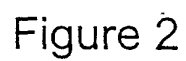
5 conferences to be easily provided. SIP messages are associated with computer software code such as Java byte code, Java applets or mobile autonomous software agents. An example of a mobile autonomous agent is a Java mobile agent. This computer software code may be contained in the body of a SIP message or an address indicating where the computer software code is located is stored in the SIP

10 message. SIP clients are arranged such that on receipt of a SIP message that has been associated with computer software code, that code is executed by a processor associated with the SIP client. For example, in the case that Java applets are contained in a SIP message these are executed by a Java Virtual Machine associated with the SIP client. If a Java mobile agent is contained in the SIP

15 message this executes on a Java Mobile Agent Virtual Machine associated with the SIP client. In one example, such computer software code must always be executed by the processor associated with the SIP client before that SIP client carries out any other actions related to the SIP message. Preferably an indicator is put into the header of a SIP message to indicate that it has been associated with computer

20 software code, and SIP clients are arranged to detect the presence of such indicators. An application programming interface is created in order that the computer software code may control the SIP client and/or any processor associated with that SIP client. In one example, computer software code is associated with SIP messages in order that a service for automatically setting up multi-media

25 conferences is provided.



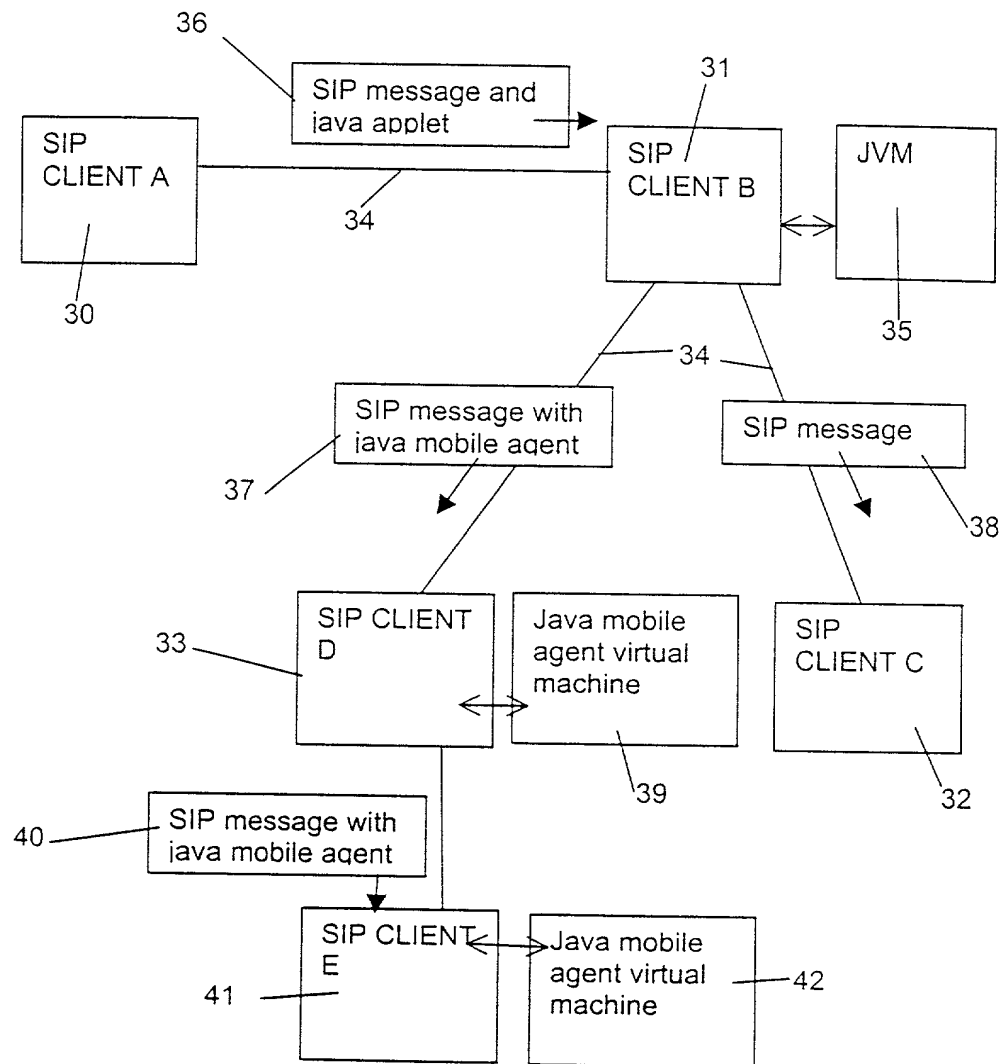


Figure 3

(Within the message body)

Figure 5

Figure 5

4/6

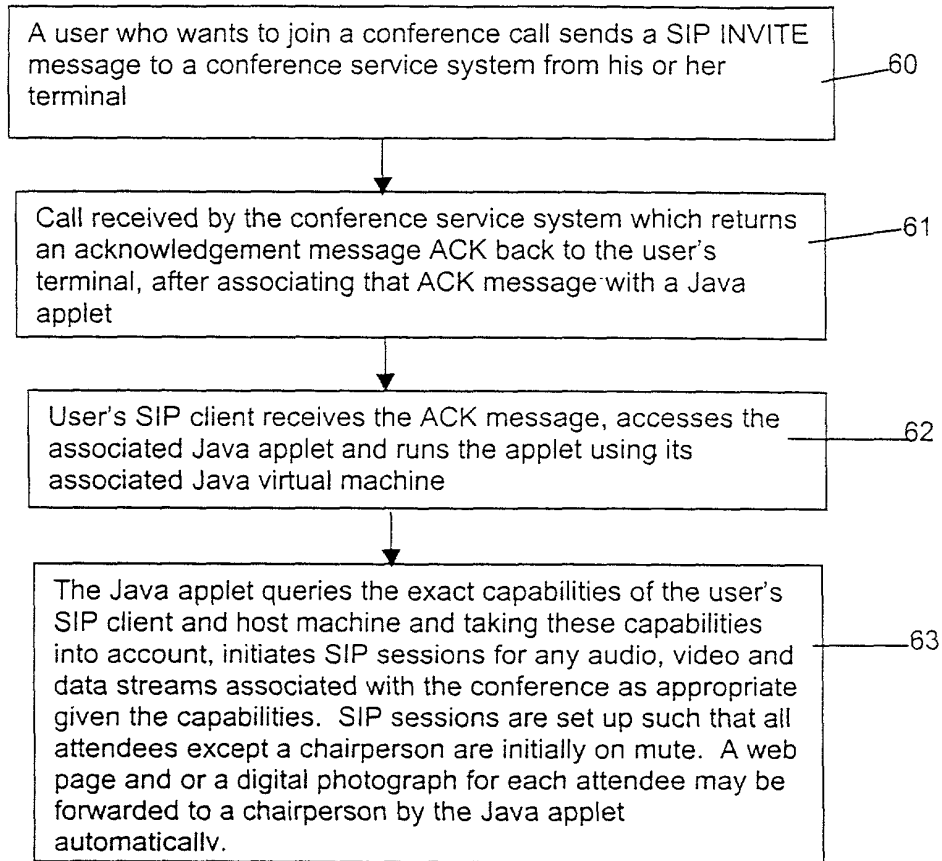


Figure 6

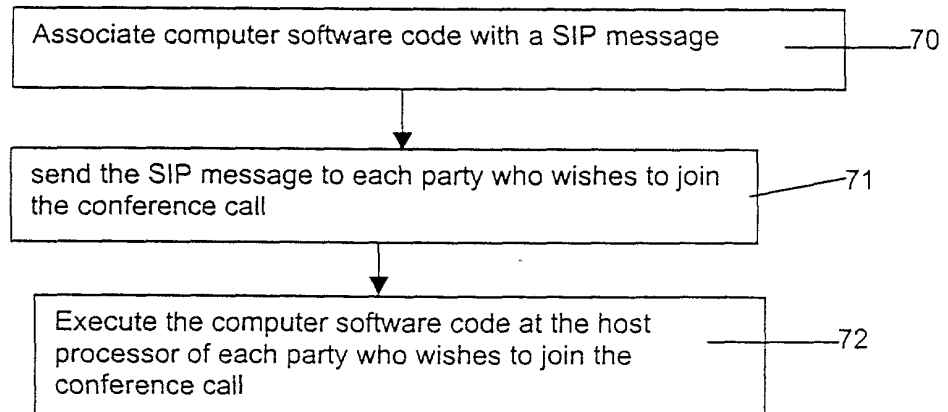


Figure 7

5/6

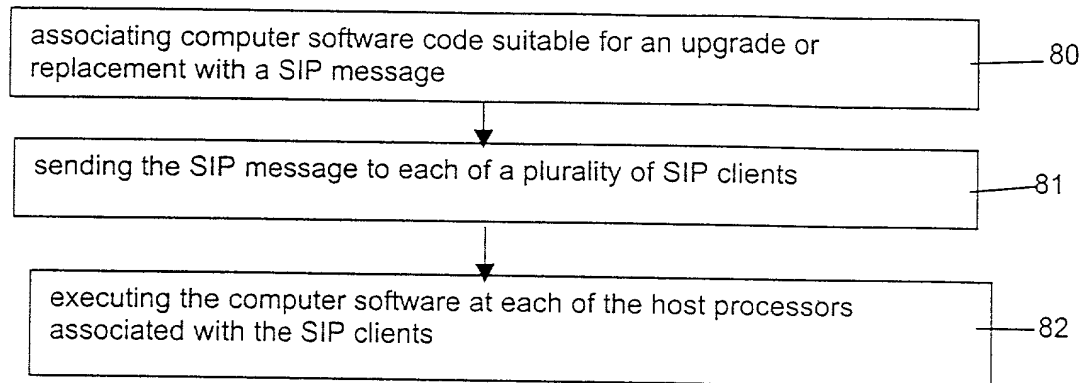


Figure 8

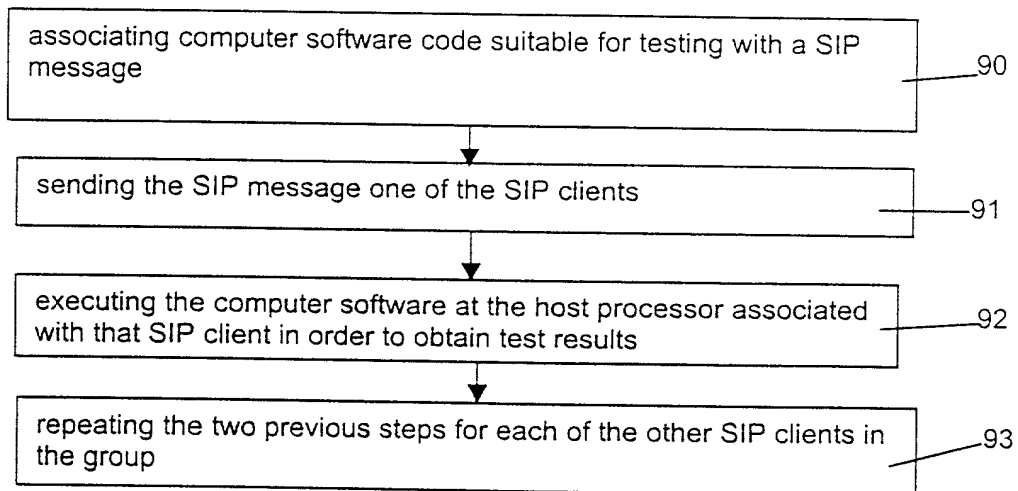


Figure 9

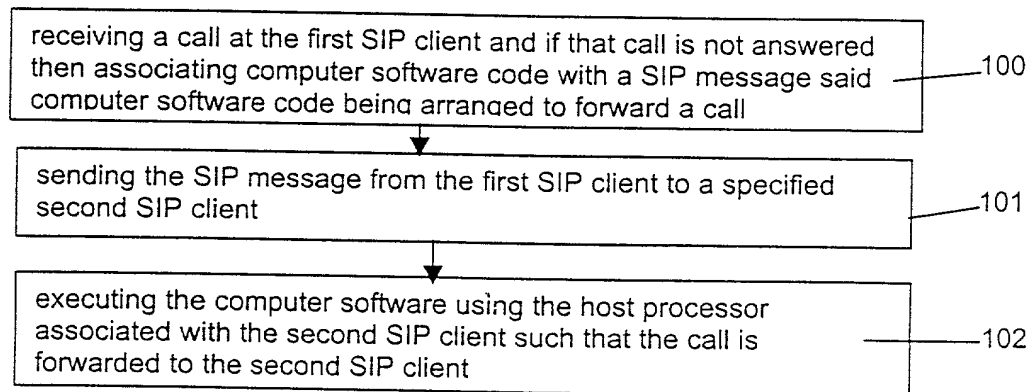


Figure 10

002020" E5B02350


```

Request = Request-Line
        * ( general-header
          | request-header
          | entity-header )
        CRLF
        [ message-body ]

```

} 40

	general-header	=	Accept		} 41
			Accept-Encoding		
			Accept-Language		
			Call-ID		
			Contact		
			CSeq		
			Date		
			Encryption		
			Expires		
			From		
			Record-Route		
			Timestamp		
			To		
			Via		
42 {	entity-header	=	Content-Encoding		
			Content-Length		
			Content-Type		
43 {	request-header	=	Authorization		< ---- This will indicate that the content type is a java applet or a Java Mobile Agent (or the URL of a location of either from where they must be retrieved).
			Contact		
			Hide		
			Max-Forwards		
			Organization		
			Priority		
			Proxy-Authorization		
			Proxy-Require		
			Route		
			Require		< ----- This will be used to indicate that Java-enhanced-SIP must be supported to process this message.
			Response-Key		
		Subject			
		User-Agent			
	response-header	=	Allow		} 44
			Proxy-Authenticate		
			Retry-After		
			Server		
			Unsupported		
			Warning		
			WWW-Authenticate		

FIGURE 4

Invention Disclosure No:11790

DECLARATION AND POWER OF ATTORNEY FOR PATENT APPLICATION

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated
below next to my name.

I believe I am the original, first and sole inventor (if only one name is listed below) or an
original, first and joint inventor (if plural names are listed below) of the subject matter
which is claimed and for which a patent is sought on the invention entitled Improved
Session Initiation Protocol (SIP), the specification of which:

X is attached hereto.

_____ was filed on _____ as

Application Serial No.

and was amended on _____ (if applicable).

I hereby state that I have reviewed and understand the contents of the above identified
specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information which is material to the examination of this
application in accordance with Title 37, Code of Federal Regulations, Section 1.56(a).

I hereby claim foreign priority benefits under Title 35, United States Code, Section 119 of
any foreign application(s) for patent or inventor's certificate listed below and have also
identified below any foreign application for patent or inventor's certificate having a filing
date before that of the application on which priority is claimed:

002020"25802560

Invention Disclosure No:11790

PRIOR FOREIGN APPLICATION(S)

<u>Country</u>	<u>Number</u>	<u>Date Filed</u>	<u>Priority Claimed</u>	
			<u>Yes</u>	<u>No</u>
_____	_____	_____		
_____	_____	_____		
_____	_____	_____		

I hereby claim the benefit under Title 35, United States Code Section 120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, Section 112, I acknowledge the duty to disclose material information as defined in Title 37, Code of Federal Regulations, Section 1.56(a) which occurred between the filing date of the prior application and the national or PCT international filing date of this application.

<u>Application Serial No.</u>	<u>Filing Date</u>	<u>Status</u>
_____	_____	
_____	_____	

And I hereby appoint Wm. Marshall Lee, Registration No. 16,853, John M. Mann, Registration No. 17,775, Thomas E. Smith, Registration No. 18,243, Dennis M. McWilliams, Registration No. 25,195, James R. Sweeney, Registration No. 18,721, William M. Lee, Jr., Registration No. 26,935, Glenn W. Ohlson, Registration No. 28,455, David C. Brezina, Registration No. 34,128, Jeffrey R. Gray, Registration No. 33,391, Timothy J. Engling, Registration No. 39,970, Gregory B. Beggs, Registration No. 19,286, Gerald S. Geren, Registration No. 24,528 and Peter J. Shakula, Registration No. 40,808 as my attorneys to prosecute this application and to transact all business in the Patent and Trademark Office connected herewith. It is requested that all communications be directed

002000"E3802550


Invention Disclosure No: 11790

to Lee, Mann, Smith, McWilliams, Sweeney & Ohlson, P.O. Box 2786, Chicago, Illinois

60690-2786, telephone number (312) 368-1300.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Full name of sole or first inventor: Michael O'Doherty

Signature 

Date 02-03-2000

Country of Residence: UK

Country of Citizenship: Ireland

Post Office and Residence Address: 33a Bollo Lane, Chiswick, London, United Kingdom

00/000 "E5802350